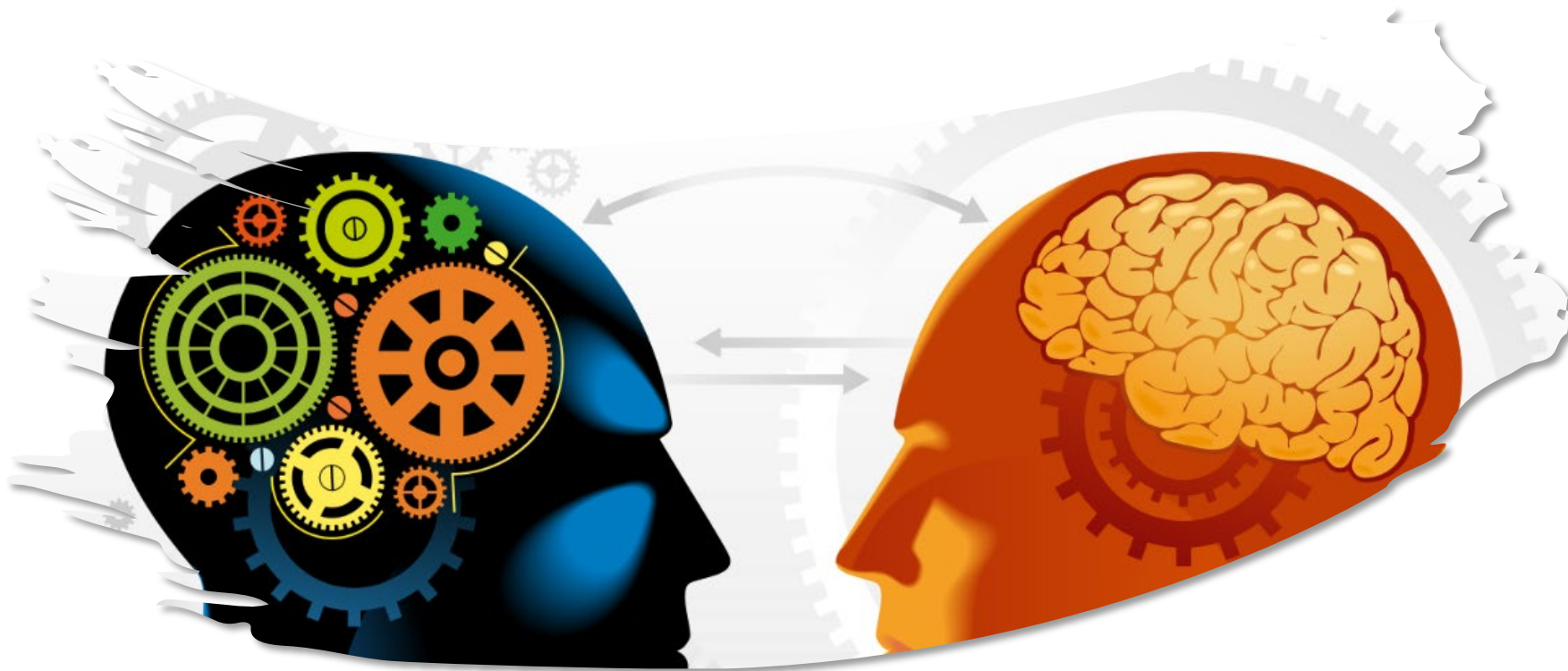




Building the Infinite Brain

COMP 690 (193)

Raghavendra Pradyumna Pothukuchi



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

✉ raghav@cs.unc.edu

Quick Review

What is pipelining?

Splitting work into many components executed independently, and passed in a chain

Why pipelining?

Increases efficiency via parallelism

*Pipelining was conceived to meet hard design goals—
IBM Stretch, ILLIAC etc.
BCI processing creates a similar need!*

What are the challenges in pipelining?

Hazards: structural, data, control

How to fix hazards?

Concurrency (structural), eager (forwarding data), eager or speculative (branch delays, prediction)

What systems design principles does pipelining touch?

Tradeoff simplicity for efficiency, leveraging parallelism through regrouping



For Today

- Quick review
- **Dynamic scheduling**



Search for Efficiency

Fixed by forwarding

ADD R1 $\leftarrow$ R2, R3
SUB R4 $\leftarrow$ R1, R5
AND R6 $\leftarrow$ R1, R7
OR R8 $\leftarrow$ R1, R9
XOR R10 $\leftarrow$ R1, R11

Unavoidable stall

LD R1 $\leftarrow$ 0 (R2)
ADD R2 $\leftarrow$ R1, R3
AND R6 $\leftarrow$ R1, R7
OR R8 $\leftarrow$ R1, R9
XOR R10 $\leftarrow$ R1, R11

What happens with long latency instructions?

FDIV F0 $\leftarrow$ F2, F4 (4)
FADD F6 $\leftarrow$ F0, F2 (2)
FMUL F4 $\leftarrow$ F2, F4 (3)

	1	2	3	4	5	6	7	8	9	10
FDIV	IF	ID	EX.d	EX.d	EX.d	EX.d	MEM	WB		
FADD		IF	ID				EX.a	EX.a	MEM	WB
FMUL			IF				ID	EX.m	EX.m	EX.m



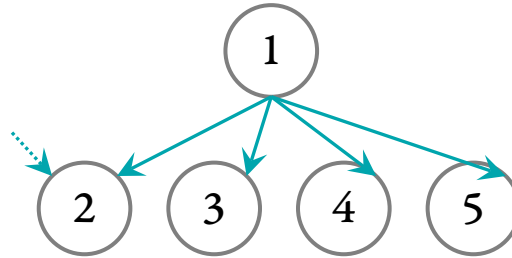
Search for Efficiency: “*Concurrency*”

1 **ADD** R1 ← R2, R3
2 **SUB** R4 ← R1, R5
3 **AND** R6 ← R1, R7
4 **OR** R8 ← R1, R9
5 **XOR** R10 ← R1, R11

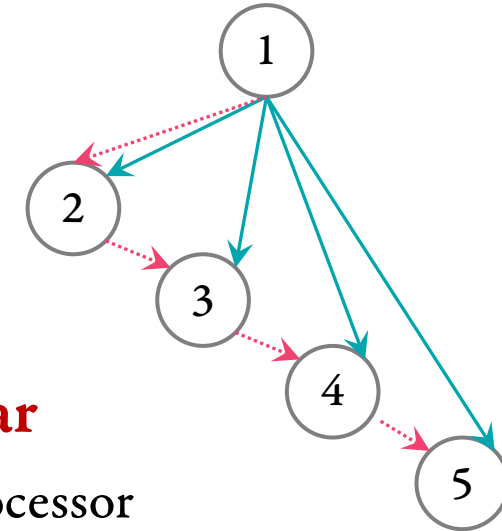
Notion of **state**:
“*being and becoming*”

More later

What you see



What you get



Superscalar

Multiple *issue* processor

Not scalar (1-issue) or vector (lockstep multi-issue)

- Increase fetch and decode widths
- Replicate functional units (ALUs)
- Complex dependency and forwarding (N^2), register and memory access



Maintaining Correctness

FDIV $\underline{F0} \leftarrow F2, F4 \quad (4)$

FADD $F6 \leftarrow \underline{F0}, F2 \quad (2)$

FMUL $F4 \leftarrow F2, F4 \quad (3)$

2-wide superscalar

	1	2	3	4	5	6	7	8	9	10
FDIV	IF	ID	EX.d	EX.d	EX.d	EX.d	MEM	WB		
FADD	IF	ID	EX.a	EX.a	MEM	WB				
FMUL		IF	ID	EX.m	EX.m	EX.m	MEM	WB		

RAW hazards

WAR and WAW hazards!

Not a new problem with superscalar—
exists even in basic pipeline with variable latency instructions

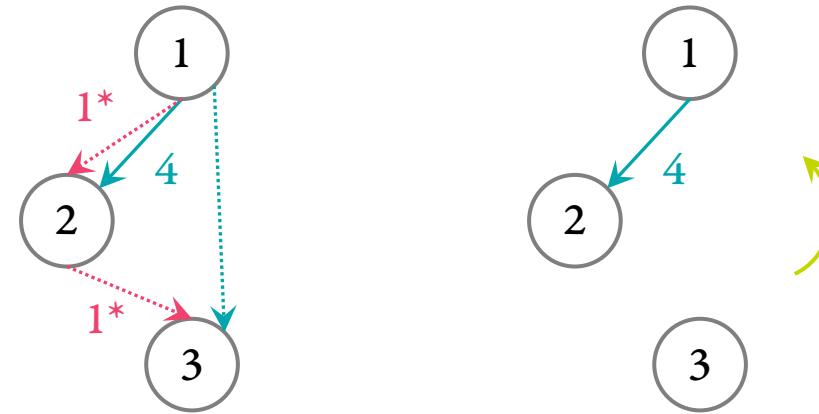
How to solve?

Stalls and cascade of stalls?



Search for Efficiency: More “*Concurrency*”

FDIV $\underline{F0} \leftarrow F2, F4$ (4)
FADD $\underline{F6} \leftarrow \underline{F0}, F2$ (2)
FMUL $F4 \leftarrow \underline{F2}, F4$ (3)



Reordering

Scoreboarding

Tomasulo's algorithm

Reorder buffer

Recover (or identify) the true dataflow in the instruction stream



Scoreboarding



CDC 6600, 1964

"Parallel Operation in the Control Data 6600"

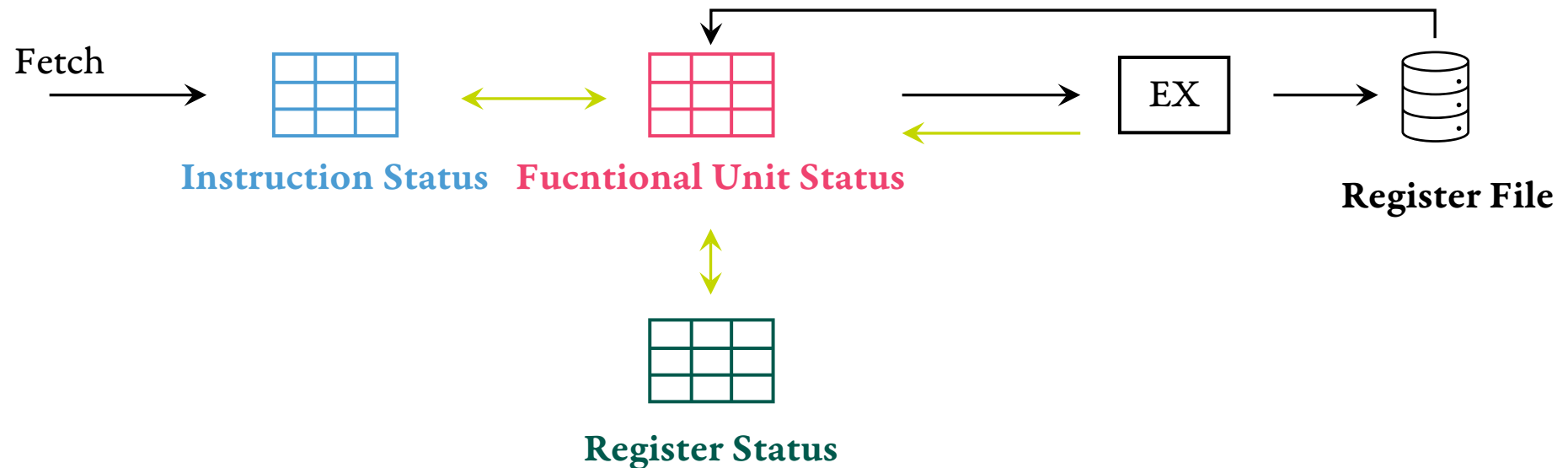
James E Thornton, Control Data Corporation (CDC)

Skip MEM (for FP)

Dispatch: decode operation and operands
Issue: read operands

Fetch → Issue → Read → Execute → Write-back

Dispatch → Issue



Scoreboarding Data Structures

Instruction Status					Register Status	
Inst	Dispatch	Issue	Execute	Write-back	Register	Unit

Functional Unit Status									
Unit	Busy	Op	Dest	R_{i1}	R_{i2}	$Src(R_{i1})$	$Src(R_{i2})$	$Ready(R_{i1})$	$Ready(R_{i2})$



Scoreboarding Example

				Instruction Status					Register Status	
				Inst	Dispatch	Issue	Execute	Write-back	Register	Unit
1	LD	F0	← 34 (R1)	1					F0	
2	MUL	F4	← F0, F2	2					F2	
3	ST	F4	→ 45 (R1)	3					F4	
4	ADD	R1	← R1, 4	4					R1	
5	LD	F0	← 34 (R1)	5						
6	MUL	F4	← F0, F2	6						
7	ST	F4	→ 45 (R1)	7						

Checks are expressed formally
and implemented, e.g.,
 $\text{Busy}[\text{Unit}] \wedge \neg \text{Dispatch} \dots$

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld									
St									
Int									
Mul1									
Mul2									



Scoreboarding Example: T₁

1	LD	F0	←	34 (R1)
2	MUL	F4	←	F0, F2
3	ST	F4	→	45 (R1)
4	ADD	R1	←	R1, 4
5	LD	F0	←	34 (R1)
6	MUL	F4	←	F0, F2
7	ST	F4	→	45 (R1)

Instruction Status				
Inst	Dispatch	Issue	Execute	Write-back
1	1			
2				
3				
4				
5				
6				
7				

Register Status	
Register	Unit
F0	Ld
F2	
F4	
R1	

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1				1	
St									
Int									
Mul1									
Mul2									



Scoreboarding Example: T₂

1	LD	F0	←	34 (R1)
2	MUL	F4	←	F0, F2
3	ST	F4	→	45 (R1)
4	ADD	R1	←	R1, 4
5	LD	F0	←	34 (R1)
6	MUL	F4	←	F0, F2
7	ST	F4	→	45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1		
2	1			
3				
4				
5				
6				
7				

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul1
R1	

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1				1	
St									
Int									
Mul1	1	mul	F4	F0	F2	Ld		0	1
Mul2									



Scoreboarding Example: T₃

1	LD	F0	←	34 (R1)
2	MUL	F4	←	F0, F2
3	ST	F4	→	45 (R1)
4	ADD	R1	←	R1, 4
5	LD	F0	←	34 (R1)
6	MUL	F4	←	F0, F2
7	ST	F4	→	45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	
2	1	! RAW Stall		
3	1			
4				
5				
6				
7				

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul1
R1	

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1					
St	1	st		F4	R1	Mul1		0	
Int									
Mul1	1	mul	F4	F0	F2	Ld		0	1
Mul2									



Scoreboarding Example: T₄

1	LD	F0	←	34 (R1)
2	MUL	F4	←	F0, F2
3	ST	F4	→	45 (R1)
4	ADD	R1	←	R1, 4
5	LD	F0	←	34 (R1)
6	MUL	F4	←	F0, F2
7	ST	F4	→	45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1		
3	1	! RAW Stall		
4	1			
5				
6				
7				

Register Status	
Register	Unit
F0	
F2	
F4	Mul1
R1	Int

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld									
St	1	st		F4	R1	Mul1		0	
Int	1	add	R1	R1				1	
Mul1	1	mul	F4	F0	F2			1	1
Mul2									



Scoreboarding Example: T₅

1	LD	F0	←	34 (R1)
2	MUL	F4	←	F0, F2
3	ST	F4	→	45 (R1)
4	ADD	R1	←	R1, 4
5	LD	F0	←	34 (R1)
6	MUL	F4	←	F0, F2
7	ST	F4	→	45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	
3	1	!		
4	1	1		
5	1			
6				
7				

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul1
R1	Int

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1		Int			
St	1	st		F4	R1	Mul1		0	
Int	1	add	R1	R1				1	
Mul1	1	mul	F4	F0	F2				
Mul2									



Scoreboarding Example: T₆

1	LD	F0	←	34 (R1)
2	MUL	F4	←	F0, F2
3	ST	F4	→	45 (R1)
4	ADD	R1	←	R1, 4
5	LD	F0	←	34 (R1)
6	MUL	F4	←	F0, F2
7	ST	F4	→	45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	
3	1	!		
4	1	1	1	
5	1	! RAW Stall		
6	! WAW Stall			
7				

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul1
R1	Int

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1		Int		0	
St	1	st		F4	R1	Mul1		0	
Int	1	add	R1	R1				1	
Mul1	1	mul	F4	F0	F2				
Mul2									



Scoreboarding Example: T₇

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	
3	1	!		
4	1	1	1	! WAR Stall
5	1	!		
6	!			
7				

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul1
R1	Int

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1		Int		0	
St	1	st		F4	R1	Mul1		0	
Int	1	add	R1	R1				1	
Mul1	1	mul	F4	F0	F2				
Mul2									



Scoreboarding Example: T₈

1	LD	F0	←	34 (R1)
2	MUL	F4	←	F0, F2
3	ST	F4	→	45 (R1)
4	ADD	R1	←	R1, 4
5	LD	F0	←	34 (R1)
6	MUL	F4	←	F0, F2
7	ST	F4	→	45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	1
3	1	1		
4	1	1	1	!
5	1	!		
6	1			
7				

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul2
R1	Int

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1		Int		0	
St	1	st		F4	R1			1	
Int	1	add	R1	R1				1	
Mul1									
Mul2	1	mul	F4	F0	F2	Ld		0	



Scoreboarding Example: T₉

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	1
3	1	1	1	
4	1	1	1	1
5	1	1		
6	1			
7	! Structural Hazard Stall			

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul2
R1	

Dispatch is always in order.
Issue and Execute may not be.

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1				1	
St	1	st		F4	R1				
Int									
Mul1									
Mul2	1	mul	F4	F0	F2	Ld		0	



Scoreboarding Example: T₁₀

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	1	1	1	1
5	1	1	1	
6	1			
7	1			

Register Status	
Register	Unit
F0	Ld
F2	
F4	Mul2
R1	

Functional Unit Status									
Unit	Busy	Op	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Ready(R _{i1})	Ready(R _{i2})
Ld	1	ld	F0	R1				1	
St	1	st		F4	R1				
Int									
Mul1									
Mul2	1	mul	F4	F0	F2	Ld		0	



Scoreboarding

“Out-of-Order” scheduling

Execute instructions when operands are ready and if no dependencies exist

WAW stalls

Functional unit status conflict

WAR stalls

Instruction status conflict

Structural stalls

Functional unit status conflict

Branches hold up dispatch



Tomasulo's Algorithm



IBM System/360 Model 91, 1967

"An Efficient Algorithm for Exploiting Multiple Arithmetic Units"

Robert M Tomasulo, IBM

```

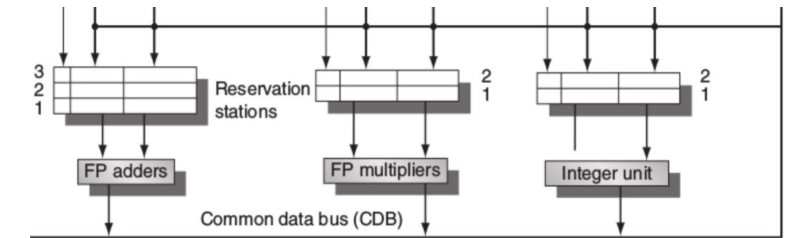
1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST  F4 → 45 (R1)
    
```

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	
3	1	!		
4	1	1	1	! WAR Stall
5	1	!		
6	!			
7				

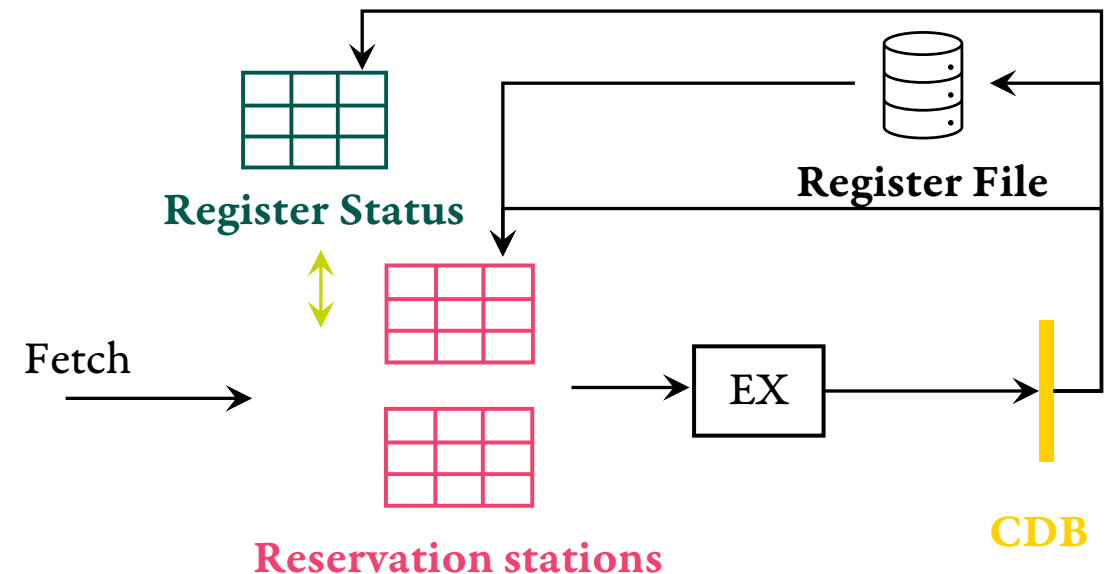
What if we version or rename registers?

Each write is to a new renamed register

Compilers use the same concept statically:
Static Single Assignment (IBM, 1980s)



Issue
Dispatch→Issue



Tomasulo's Data Structures

Reservation Station (per functional unit)							
Num	Op	Busy	R_{i1}	R_{i2}	$Src(R_{i1})$	$Src(R_{i2})$	Address

Register File	
Register	Num



For illustration

Reservation Station								
Unit	Num	Op	Busy	R_{i1}	R_{i2}	$Src(R_{i1})$	$Src(R_{i2})$	Address

Instruction Status				
Inst	Dispatch	Issue	Execute	Write-back

Tomasulo Example

1 LD F0 $\leftarrow$ 34 (R1)
2 MUL F4 $\leftarrow$ F0, F2
3 ST F4 $\rightarrow$ 45 (R1)
4 ADD R1 $\leftarrow$ R1, 4
5 LD F0 $\leftarrow$ 34 (R1)
6 MUL F4 $\leftarrow$ F0, F2
7 ST F4 $\rightarrow$ 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1				
2				
3				
4				
5				
6				
7				

Register Status	
Register	Num
F0	
F2	
F4	
R1	

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1							
St	2							
Int	3							
Mul1	4							
Mul2	5							



Tomasulo Example: T₁

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1			
2				
3				
4				
5				
6				
7				

Register Status	
Register	Num
F0	1
F2	
F4	
R1	

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1	Reg[R1]				34
St	2							
Int	3							
Mul1	4							
Mul2	5							



Tomasulo Example: T₂

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1		
2	1			
3				
4				
5				
6				
7				

Register Status	
Register	Num
F0	1
F2	
F4	4
R1	

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1	Reg[R1]				34+Reg[R1]
St	2							
Int	3							
Mul1	4	mul	1		Reg[F2]	1		
Mul2	5							



Tomasulo Example: T₃

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	
2	1	! RAW Stall		
3	1			
4				
5				
6				
7				

Register Status	
Register	Num
F0	1
F2	
F4	4
R1	

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1	Reg[R1]				34+Reg[R1]
St	2	st	1		Reg[R1]	4		45
Int	3							
Mul1	4	mul	1		Reg[F2]	1		
Mul2	5							



Tomasulo Example: T₄

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1		
3	1	! RAW Stall		
4	1			
5				
6				
7				

Register Status	
Register	Num
F0	
F2	
F4	4
R1	3

Val, 1

CDB

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1							
St	2	st	1		Reg[R1]	4		45
Int	3	add	1	Reg[R1]	4			
Mul1	4	mul	1	CDB.V	Reg[F2]			
Mul2	5							



Tomasulo Example: T₅

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	
3	1	!		
4	1	1		
5	1			
6				
7				

Register Status	
Register	Num
F0	1
F2	
F4	4
R1	3

CDB

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1			3		34
St	2	st	1		Reg[R1]	4		45
Int	3	add	1	Reg[R1]	4			
Mul1	4	mul	1		Reg[F2]			
Mul2	5							



Tomasulo Example: T₆

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	
3	1	!		
4	1	1	1	
5	1	! RAW Stall		
6	1	No WAR stall		
7				

Register Status	
Register	Num
F0	1
F2	
F4	5
R1	3

CDB

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1			3		34
St	2	st	1		Reg[R1]	4		45
Int	3	add	1	Reg[R1]	4			
Mul1	4	mul	1		Reg[F2]			
Mul2	5	mul	1		Reg[F2]	1		



Tomasulo Example: T₇

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	
3	1	!		
4	1	1	1	1 <i>No WAR stall</i>
5	1	1		
6	1			
7	! Structural Hazard Stall			

Register Status	
Register	Num
F0	1
F2	
F4	5
R1	

Val, 3

CDB

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1	CDB.V				34
St	2	st	1		Reg[R1]	4		45
Int	3							
Mul1	4	mul	1		Reg[F2]			
Mul2	5	mul	1		Reg[F2]	1		



Tomasulo Example: T₈

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Val, 4

CDB

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	1
3	1	1		
4	1	1	1	1
5	1	1	1	
6	1			
7	!			

Register Status	
Register	Num
F0	1
F2	
F4	5
R1	

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1					34+[1. R _{i1}]
St	2	st	1	CDB.V	Reg[R1]			45
Int	3							
Mul1	4							
Mul2	5	mul	1		Reg[F2]	1		



Tomasulo Example: T₉

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	1
3	1	1	1	
4	1	1	1	1
5	1	1	1	1
6	1	1		
7	!			

Register Status	
Register	Num
F0	
F2	
F4	5
R1	

Val, 1

CDB

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1							
St	2	st	1		Reg[R1]			45+[2. R _{i1}]
Int	3							
Mul1	4							
Mul2	5	mul	1	CDB.V	Reg[F2]			



Tomasulo Example: T₁₀

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Inst	Instruction Status			
	Dispatch	Issue	Execute	Write-back
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	1	1	1	1
5	1	1	1	1
6	1	1	1	
7	1			

Register Status	
Register	Num
F0	
F2	
F4	5
R1	

Reservation Station								
Unit	Num	Op	Busy	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1							
St	2	st	1		Reg[R1]	5		45
Int	3							
Mul1	4							
Mul2	5	mul	1		Reg[F2]			

CDB



Tomasulo's Algorithm

Renaming and operand buffering

WAR and WAW nuances with loads and stores

Structural stalls

Functional unit status conflict

Branches hold up dispatch

Are we at the limit?



Speculation

Predict branches and branch targets

Branch folding—zero cycle branches!

Preserve state

Being: Value of registers (architectural) and *Becoming*: the sequence of operations (for recovery)

Sometimes imprecision (“approximation”) is acceptable, e.g., program terminates



A Structure to Maintain State



Reorder Buffer, 1985

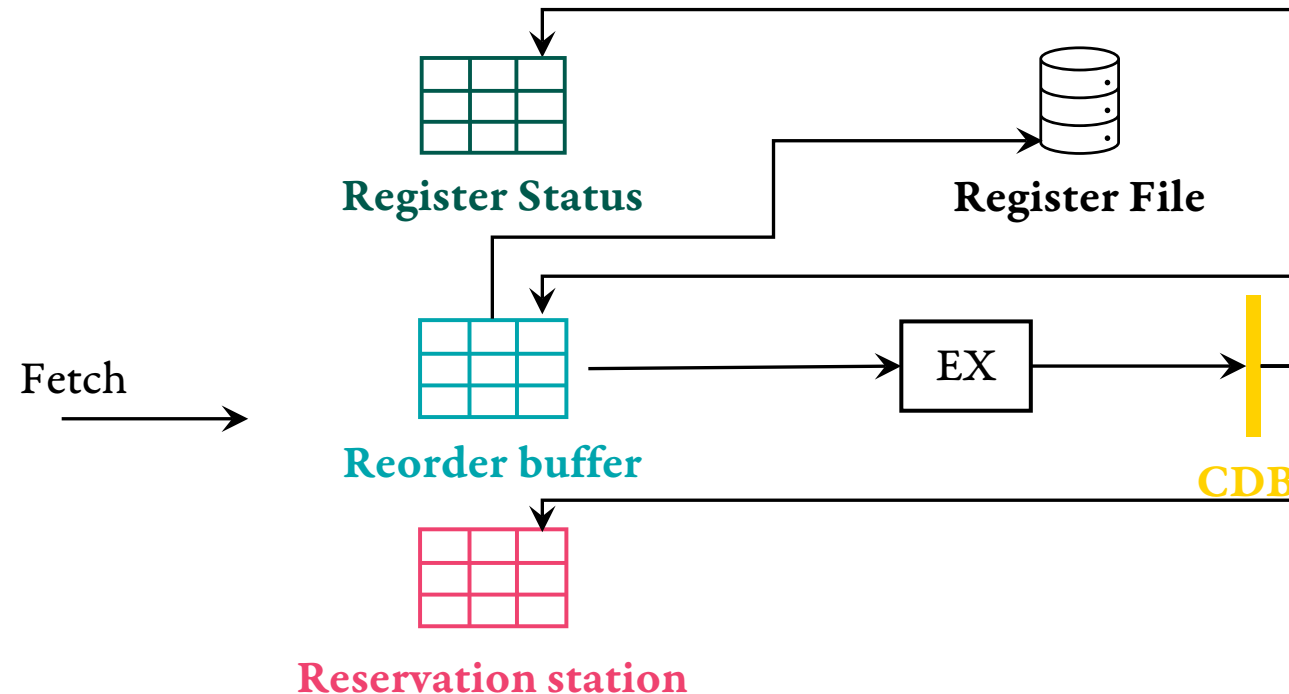
["Implementation of Precise Interrupts in Pipelined Processors"](#)

James E. Smith, Andrew R. Pleszkun

Commit is always in order

Complete vs commit

Fetch→Dispatch→Execute→Complete→Commit



Speculation Data Structures

Reorder Buffer										
Ptr	Num	Inst	Dest	Value	Ready	PC	Exception	Issue	Execute	Complete

Reservation Station									
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address

Register File	
Register	Num



Speculation Example

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1								
	2								
	3								
	4								
	5								
	6								
	7								

Register Status		
Register	Num	ROB.V
F0		
F2		
F4		
R1		

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2									
Int	3									
Mul1	4									
Mul2	5									



Speculation Example: T₁

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
ht	1	1	F0	Val[F0]					
	2								
	3								
	4								
	5								
	6								
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.1	
F2		
F4		
R1		

Reservation Station											
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address		
Ld	1	ld	1	ROB.1	Reg[R1]						34
St	2										
Int	3										
Mul1	4										
Mul2	5										



Speculation Example: T₂

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
h	1	1	F0	Val[F0]			1		
t	2	2	F4						
	3								
	4								
	5								
	6								
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.1	
F2		
F4	ROB.2	
R1		

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1	ld	1	ROB.1	Reg[R1]				34+Reg[R1]	
St	2									
Int	3									
Mul1	4	mul	1	ROB.2		Reg[F2]	ROB.1			
Mul2	5									



Speculation Example: T₃

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
h	1	1	F0	Val[F0]			1	1	
	2	2	F4				! RAW Stall		
t	3	3							
	4								
	5								
	6								
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.1	
F2		
F4	ROB.2	
R1		

Reservation Station									
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	<i>Free on execute</i>							
St	2	st	1	ROB.3		Reg[R1]	ROB.2		45
Int	3								
Mul1	4	mul	1	ROB.2		Reg[F2]	ROB.1		
Mul2	5								



Speculation Example: T₄

Val, 1

CDB

```

1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST  F4 → 45 (R1)
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
h	1	1	F0	CDB.V			1	1	1
	2	2	F4				1		
	3	3					! RAW Stall		
t	4	4	R1						
	5								
	6								
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.1	1
F2		
F4	ROB.2	
R1	ROB.4	

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2	st	1	ROB.3		Reg[R1]	ROB.2		45	
Int	3	add	1	ROB.4	Reg[R1]	4				
Mul1	4	mul	1	ROB.2	CDB.V	Reg[F2]				
Mul2	5									



Speculation Example: T₅

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
h	2	2	F4				1	1	
	3	3					!		
	4	4	R1				1		
t	5	5	F0						
	6								
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.5	
F2		
F4	ROB.2	
R1	ROB.4	

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1	ld	1	ROB.5			ROB.4		34	
St	2	st	1	ROB.3		Reg[R1]	ROB.2		45	
Int	3	add	1	ROB.4	Reg[R1]	4				
Mul1	4									
Mul2	5									



Speculation Example: T₆

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
h	2	2	F4				1	1	
	3	3					!		
	4	4	R1				1	1	
	5	5	F0				!	RAW Stall	
t	6	6	F4						
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.5	
F2		
F4	ROB.6	
R1	ROB.4	

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1	ld	1	ROB.5			ROB.4		34	
St	2	st	1	ROB.3		Reg[R1]	ROB.2		45	
Int	3									
Mul1	4	mul	1	ROB.6		Reg[F2]	ROB.5			
Mul2	5									



Speculation Example: T₇

Val, 4

CDB

```

1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST  F4 → 45 (R1)
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
h	2	2	F4				1	1	
	3	3					!		
	4	4	R1	CDB.V			1	1	1
	5	5	F0				1		
t	6	6	F4						
	7	! Structural Hazard Stall							

Register Status

Register	Num	ROB.V
F0	ROB.5	
F2		
F4	ROB.6	
R1	ROB.4	1

Reservation Station

Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1	ld	1	ROB.5	CDB.V				34
St	2	st	1	ROB.3		Reg[R1]	ROB.2		45
Int	3								
Mul1	4	mul	1	ROB.6		Reg[F2]	ROB.5		
Mul2	5								



Speculation Example: T₈

Val, 2

CDB

```

1 LD  F0  ← 34 (R1)
2 MUL F4  ← F0, F2
3 ST  F4  → 45 (R1)
4 ADD R1  ← R1, 4
5 LD  F0  ← 34 (R1)
6 MUL F4  ← F0, F2
7 ST  F4  → 45 (R1)
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
h	2	2	F4				1	1	1
	3	3					1		
	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	
t	6	6	F4						
	7	! Structural Hazard Stall							

Register Status		
Register	Num	ROB.V
F0	ROB.5	
F2		
F4	ROB.6	
R1	ROB.4	1

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2	st	1	ROB.3	CDB.V	Reg[R1]			45	
Int	3									
Mul1	4	mul	1	ROB.6		Reg[F2]	ROB.5			
Mul2	5									



Speculation Example: T₉

Val,5

CDB

```

1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST  F4 → 45 (R1)
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
h	3	3					1	1	
	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	1
	6	6	F4				1		
t	7	7							

Register Status		
Register	Num	ROB.V
F0	ROB.5	1
F2		
F4	ROB.6	
R1	ROB.4	1

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2	1	1	ROB.7		ROB[4]	ROB.6			
Int	3									
Mul1	4	mul	1	ROB.6	CDB.V	Reg[F2]				
Mul2	5									



Speculation Example: T₁₀

CDB

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
h	3	3					1	1	1
	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	1
	6	6	F4				1	1	
t	7	7					! RAW Stall		

Register Status		
Register	Num	ROB.V
F0	ROB.5	1
F2		
F4	ROB.6	
R1	ROB.4	1

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2	1	1	ROB.7		ROB[4]	ROB.6			
Int	3									
Mul1	4									
Mul2	5									



Speculation Example: T₁₁

CDB

1 LD F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD F0 ← 34 (R1)
6 MUL F4 ← F0, F2
7 ST F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
	3	3					1	1	1
h	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	1
	6	6	F4				1	1	
t	7	7					!		

Register Status		
Register	Num	ROB.V
F0	ROB.5	1
F2		
F4	ROB.6	
R1	ROB.4	1

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2	1	1	ROB.7		ROB[4]	ROB.6			
Int	3									
Mul1	4									
Mul2	5									



Squash Example: T₉

Val,5

CDB

```

1 LD  F0  ← 34 (R1)
2 MUL F4  ← F0, F2
3 ST  F4  → 45 (R1)
4 ADD R1  ← R1, 4
5 LD  F0  ← 34 (R1)
6 MUL F4  ← F0, F2
7 ST  F4  → 45 (R1)
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
h	3	3				1 !	1	1	
	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	1
	6	6	F4				1		
t	7	7							

Register Status

Register	Num	ROB.V
F0	ROB.5	1
F2		
F4	ROB.6	
R1	ROB.4	1

Reservation Station

Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1								
St	2	1	1	ROB.7		ROB.3	ROB.6		
Int	3								
Mul1	4	mul	1	ROB.6	CDB.V	Reg[F2]			
Mul2	5								



Squash Example: T₁₀

CDB

```

1 LD  F0  ← 34 (R1)
2 MUL F4  ← F0, F2
3 ST   F4  → 45 (R1)
4 ADD R1  ← R1, 4
5 LD   F0  ← 34 (R1)
6 MUL F4  ← F0, F2
7 ST   F4  → 45 (R1)
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
h	3								
	4								
	5								
	6								
	7								

Register Status		
Register	Num	ROB.V
F0		
F2		
F4		
R1		

Reservation Station											
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address		
Ld	1										
St	2										
Int	3										
Mul1	4										
Mul2	5										



Squash Example: T₁₁

CDB

1 **LD** F0 ← 34 (R1)
 2 **MUL** F4 ← F0, F2
 3 **ST** F4 → 45 (R1)
 4 **ADD** R1 ← R1, 4
 5 **LD** F0 ← 34 (R1)
 6 **MUL** F4 ← F0, F2
 7 **ST** F4 → 45 (R1)

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
ht	3	3							
	4								
	5								
	6								
	7								

Register Status		
Register	Num	ROB.V
F0		
F2		
F4		
R1		

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2	st	1	ROB.3	Reg[R4]	Reg[R1]				
Int	3									
Mul1	4									
Mul2	5									



Branch Example: T

```

1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 BEQZ R3 Label
7 ADD R1 ← R1, 4
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
h	3	3					1	1	
	4	4	R1	Val[R1]			1	1	1
t	5	5	F0				1	1	1
	6								
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.5	1
F2		
F4		
R1	ROB.4	1
R3		

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2	1	1	ROB.7		ROB.3	ROB.6			
Int	3									
Mul1	4									
Mul2	5									



Branch Example: T_{+1}

```

1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 BEQZ R3 Label
7 ADD R1 ← R1, 4
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
h	3	3					1	1	1
	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	1
t	6	6							
	7								

Register Status		
Register	Num	ROB.V
F0	ROB.5	1
F2		
F4		
R1	ROB.4	1
R3		

Reservation Station									
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1								
St	2								
Int	3	br	1		Reg[R3]				
Mul1	4								
Mul2	5								



Branch Example: T_{+2}

```

1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 BEQZ R3 Label
7 ADD R1 ← R1, 4
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
	3	3					1	1	1
h	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	1
t	6	6					1		
	7	!							

Register Status		
Register	Num	ROB.V
F0	ROB.5	1
F2		
F4		
R1	ROB.4	1
R3		

Reservation Station									
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address
Ld	1								
St	2								
Int	3	br	1		Reg[R3]				
Mul1	4								
Mul2	5								



Branch Example: T₊₃

```
1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 BEQZ R3 Label
7 ADD R1 ← R1, 4
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
	3	3					1	1	1
	4	4	R1	Val[R1]			1	1	1
h	5	5	F0				1	1	1
	6	6					1	1	
t	7	7	R1						

Register Status		
Register	Num	ROB.V
F0	ROB.5	1
F2		
F4		
R1		
R3		

Reservation Station										
Unit	Num	Op	Busy	Dest	R _{i1}	R _{i2}	Src(R _{i1})	Src(R _{i2})	Address	
Ld	1									
St	2									
Int	3	ld	1	ROB.7	Reg[R1]	4				
Mul1	4									
Mul2	5									



Branch Example: T_{+4}

```

1 LD  F0 ← 34 (R1)
2 MUL F4 ← F0, F2
3 ST  F4 → 45 (R1)
4 ADD R1 ← R1, 4
5 LD  F0 ← 34 (R1)
6 BEQZ R3 Label
7 ADD R1 ← R1, 4
    
```

Reorder Buffer									
Ptr	Num	Inst	Dest	Value	PC	Exception	Issue	Execute	Complete
	1	1	F0	Val[F0]			1	1	1
	2	2	F4				1	1	1
	3	3					1	1	1
	4	4	R1	Val[R1]			1	1	1
	5	5	F0				1	1	1
h	6	6					1	1	
t	7	7	R1				1		

Register Status		
Register	Num	ROB.V
F0		
F2		
F4		
R1		
R3		

Reservation Station										
Unit	Num	Op	Busy	Dest	R_{i1}	R_{i2}	$Src(R_{i1})$	$Src(R_{i2})$	Address	
Ld	1									
St	2									
Int	3	add	1	ROB.7	Reg[R1]	4				
Mul1	4									
Mul2	5									



Speculation

Self study: Branch prediction,
Software methods for ILP like VLIW, pipelining

Structural stalls

Branches no longer holdup dispatch

Alternative to ROB supplied values: physical register file and a register alias table (RAT)

ROB still needed for ordering, and has old and new register maps for recovery

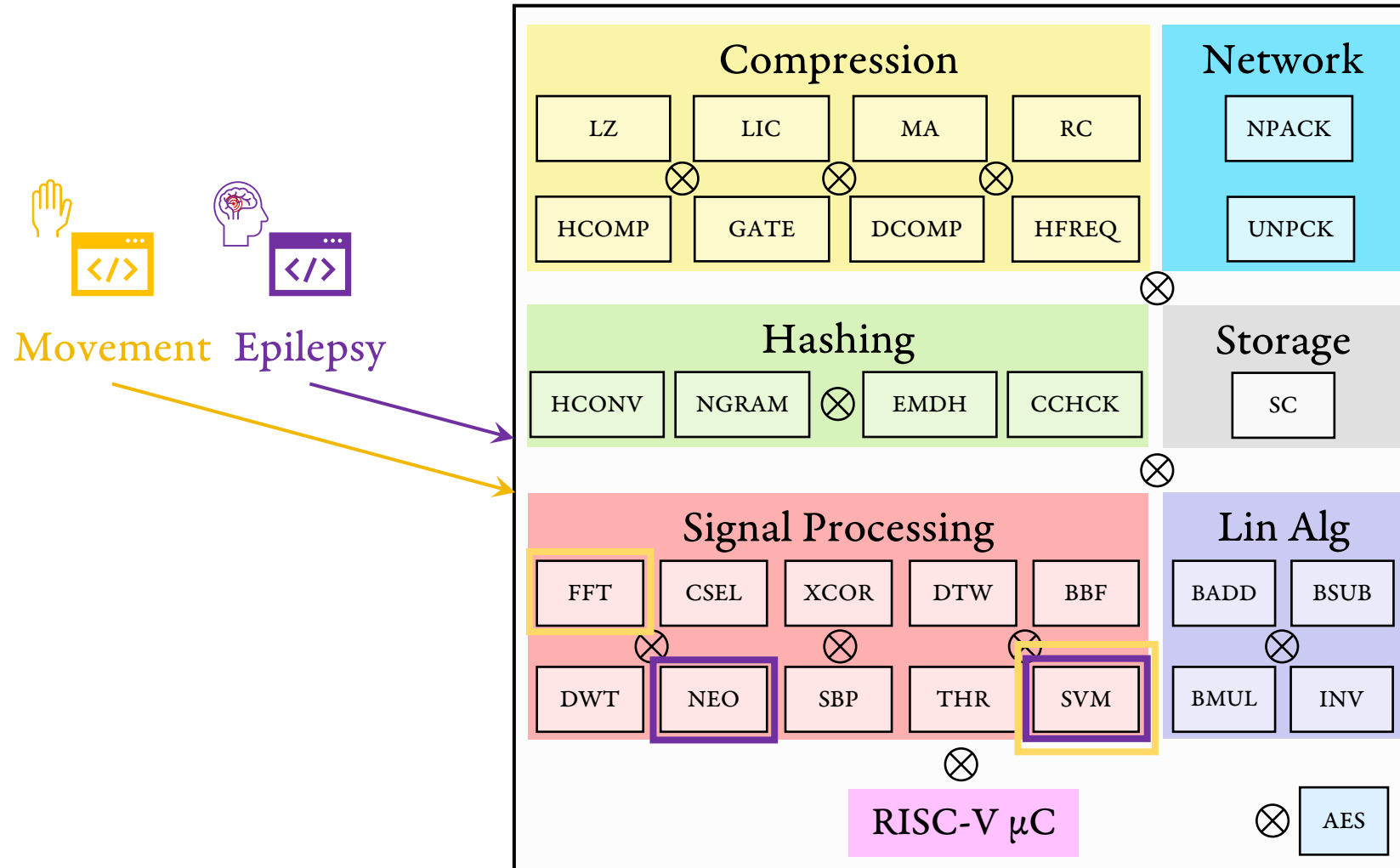
Concurrency we targeted: Instruction Level Parallelism (ILP)

Alternative or complementary: software methods—unrolling, software pipelining



Example from a BCI Processor (SCALO)

A different kind of pipeline: Dataflow—work flows through a directed acyclic graph (DAG)



Choice of PEs

Common kernels (like common subgraphs)

Pipeline scheduling

Software managed pipelining

No hardware mitigation for any hazards!

Simplicity vs Efficiency

Takeaways

Homework 2 will be released on 9/17,
due 9/30

Why dynamic scheduling?

To be resource efficient in exploiting instruction level parallelism

What are hardware methods for dynamic scheduling?

Scoreboarding, Tomasulo's algorithm, speculation

What systems principles does dynamic scheduling involve?

Eager, speculative, and concurrency

What are some other methods to exploit ILP?

Software pipelining, unrolling, which target static ILP



Image Credits (Educational, Fair Use)

- Title image: VLADGRIN, https://www.istockphoto.com/vector/human_-machine-gm147409511-16840728 (Educational fair use)
- Infinite brain: Science wonder stories, May 1930, Illustrator: Frank R Paul, Editor: Hugo Gernsback
- Brain color, ICs, cloud server, black rat: No attribution required (Hiclipart)
- Hand with spoon: public domain freepng
- Signals: <https://www.nature.com/articles/nrn3724>
- Thought clouds: F. Willett et al./*Nature* 2021/Erika Woodrum, <https://med.stanford.edu/neurosurgery/news/2022/bci-award>. <https://www.the-scientist.com/news-opinion/brain-computer-interface-user-types-90-characters-per-minute-with-mind-68762>
- Picture of scientists: <https://www.cs.auckland.ac.nz/~brian/rutherford8.html> (original: Pierre de Latil), Bush (Carnegie Science), Others (Wikipedia, National Academies, IEEE, and university profile images)
- Flowchart: Pause08 – flaticon.com
- Digital brain: Smashicons – flaticon.com
- Server rack: upklyak – freepik.com
- Quantum processor icons created by Paul J. - Flaticon
- Arm, Lotus: Adobe stock
- Quantum processor: Rigetti computing
- Images of implanted users: Top: Case Western Reserve University (<https://thedaily.case.edu/man-quadruplegia-employs-injury-bridging-technologies-move-just-thinking/>), Bottom: Jan Scheuermann (University of Pittsburgh/UPMC; <https://www.upmc.com/media/news/bci-press-release-chocolate>)
- Images of wearable BCIs: Cognixion, NextMind
- Types of BCIs: “Brain–computer interfaces for communication and rehabilitation,
- Illustrative BCI: Neuralink
- Electrodes: “Electrochemical and electrophysiological considerations for clinical high channel count neural interfaces”, Vatsyayan et al.
- Form factors: Neuropace, Medtronic, Bloomberg, “Fully Implanted Brain–Computer Interface in a Locked-In Patient with ALS” by Vansteensel et al., Blackrock Neurotech
- Jose Delgado’s video: Online, various sources (CNN, Youtube)
- Video of Kennedy and Ramsey: Online, various sources (Youtube, Neural signals)
- Code snippet inspiration: ECE 252 slides at Duke (Dan Sorin et al.)

Logos, trademarks are all properties of respective owners

Not to be shared outside the course

